FTN-M080: D5 - Japanese Unexamined Patent Publication No. 10-301831

[0020]
Embodiment

The embodiment of the present invention will be described below.

Embodiment 1: Fig. 1 shows a constitution of a network including a computer system of a first embodiment of the present invention. Numeral 1 shows a client computer for executing a program. Numeral 2 shows a server computer for transferring the program to the client computer 1. Numeral 3 shows a network for connecting the client computer 1 and the server computer 2. Numeral 4 shows a program loading and executing means, for loading and executing a program, which is provided in the client computer 1. Numeral 5 shows a program transferring means, for transferring a program, which is provided in the server computer 2. Numeral 6 shows a cache control means for controlling caches 8, 11, which is included in the program loading and executing means 4. Numeral 7 shows a non-volatile storage apparatus including the caches 8, 11, which is connected to the client computer 1. Numeral 8 shows a conventionally used normal cache (a first cache) which is provided in the non-volatile storage apparatus 7. Numeral 9 shows a non-volatile storage apparatus connected to the server computer 2. Numeral 10 shows a program stored in the non-volatile storage apparatus 9. Numeral 11 shows a permanent cache (a second cache) for storing data until a user deletes it. The client computer 1 and the non-volatile storage apparatus 7 constitute a computer system of the present invention.

[0021]

Fig. 2 shows a flowchart of operations of the program

loading and executing means 4. The operations of the program loading and executing means 4 will be discussed below with reference to the flowchart shown in Fig. 2.

[0022]

If the user specifies the program to be stored in the permanent cache 11 and inputs a starting order for storing it in the computer system, the program loading and executing means 4 firstly determines whether the program file is stored in the permanent cache 11, by accessing the cache controlling means 6 (step ST1). If the program file is stored therein, the program file is loaded using the cache controlling means 6 (step ST2). If the program file is not stored therein, the program loading and executing means 4 also determines whether the program file is stored in the normal cache 8, by accessing the cache controlling means 6 (step ST3).

[0023]

If the program file is stored in the normal cache 8, the program file is loaded from the normal cache 8 (step ST5), and if the program file is not stored therein, the program file is downloaded, using the program transferring means 5, through the network 3 (step ST4).

[0024]

When the program file is loaded through the normal cache 8 or the program transferring means 5, the program file is checked to determine whether it has been specified, by the user, to be stored in the permanent cache or is a file which has been loaded due to a need for the program file specified by the user to be stored (step ST6). If the program file is specified, by the user, to be stored or the program file is a file which is necessary for the program file specified, by the user, to be stored, the program file

is stored in the permanent cache 11 through the cache controlling means 6 (step ST 7), and if the program file is not specified, by the user, to be stored or the program file is not a file which is necessary for the program file specified, by the user, to be stored, the program file is stored in the normal cache 8 (step ST8). When the program file is stored in the normal cache 8, a countermeasure for overflow of cache is conducted in accordance with the cache controlling policy of the cache controlling means 6. For example, a file, which has not been used for the longest time and is the oldest file among those stored in the cache, is deleted from the normal cache 8, and a new file is stored in the cache.

[0025]

After file loading and the cache process are finished, the program is executed and a dependency relationship analysis is performed, so as to analyze the relationship between the dependent and to-be-dependent of the file which is necessary for the program file when executing (step ST9). If the file, which is dynamically necessary and is to be loaded for the dependency relationship analysis and execution of the file (step ST10), does not exist, the cache process ends. Otherwise, the process returns back to the cache check and the loading step (step ST1).

[0026]

As mentioned above, according to the first embodiment, if the program file cannot be downloaded through the program transferring means 5, in other words, the client computer 1 is for a mobile use, etc., and the program is not stored in the normal cache 8 and the program cannot be loaded from the normal cache 8, the effect for executing the program can be obtained by loading the program file from the permanent cache 11 because the program file

specified by the user and the file which is necessary for the specified program file are stored in the permanent cache 11.

If the program is stored in the permanent cache 11, whether the program file which is specified to be stored by the user or the file which is necessary for the program file is determined. Therefore, only the program file which the user wishes to be stored is stored in the permanent cache 11, and the effect that the cache capacity cannot be redundantly consumed can be obtained. Furthermore, if the program is provided as modules and constituted by a plurality of files, the user can store the necessary file groups without specifying the name of the program precisely, using the dependency relationship analysis function of the program loading and executing means. To update the file groups in the permanent cache 11, similar steps, in the storing process, should be conducted.

D5

(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号

特開平10－301831

(43)公開日　平成10年(1998)11月13日

(51)Int.Cl.[6]	識別記号	ＦＩ	
Ｇ０６Ｆ　12/00	５４７	Ｇ０６Ｆ　12/00	５４７Ａ
	５１４		５１４Ａ

審査請求　未請求　請求項の数９　ＯＬ　（全 20 頁）

(21)出願番号　特願平9－105011

(22)出願日　平成９年(1997)４月22日

(71)出願人　000006013
三菱電機株式会社
東京都千代田区丸の内二丁目２番３号

(72)発明者　岡田　英明
東京都千代田区丸の内二丁目２番３号　三菱電機株式会社内

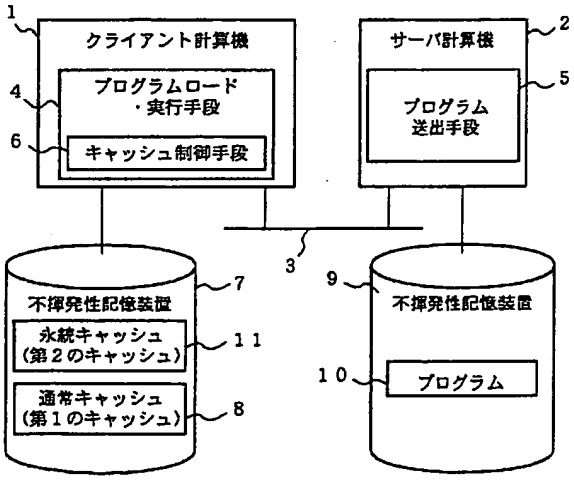
(74)代理人　弁理士　田澤　博昭　（外１名）

(54)【発明の名称】　計算機システム

(57)【要約】

【課題】　キャッシュが溢れ、プログラムの内容がキャッシュから消去されてしまうような場合、プログラムを再ロードできない場合がある。

【解決手段】　プログラムファイルを通常キャッシュに一時的に保存すると共に、永続キャッシュを不揮発性記憶装置中に設け、少なくともユーザの指定するプログラムファイルを保存し、ユーザの消去の指示があるまでこのプログラムファイルを保存し続ける。



1

【特許請求の範囲】

【請求項１】　プログラムファイルを外部から取り込み、取り込んだプログラムを実行するクライアント計算機と、該クライアント計算機が前記プログラムを実行するために必要なファイルを記録する不揮発性記憶装置とを有する計算機システムにおいて、

前記プログラムファイルを一時的に保存する第１のキャッシュと、

前記不揮発性記憶装置中に設けられ、少なくともユーザの指定するプログラムファイルを保存し、ユーザの消去の指示があるまで該プログラムファイルを保存し続ける第２のキャッシュとを備えたことを特徴とする計算機システム。

【請求項２】　第１のキャッシュ中へのプログラムファイルの保存の事実と、保存されたプログラムファイルの依存関係とを記録するキャッシュ履歴簿を不揮発性記憶装置中に設けたことを特徴とする請求項１記載の計算機システム。

【請求項３】　プログラムファイルの依存関係解析機能を持ち、プログラムをロード、キャッシュするプログラムロード・キャッシュ手段をクライアント計算機に設けたことを特徴とする請求項１または請求項２記載の計算機システム。

【請求項４】　ユーザから削除するように指示のあったファイル群を第２のキャッシュから削除し、第１のキャッシュへコピーするキャッシュ移動手段をクライアント計算機に設けたことを特徴とする請求項１から請求項３のうちのいずれか１項記載の計算機システム。

【請求項５】　第２のキャッシュ中のファイル群の削除履歴簿を不揮発性記憶装置中に設けたことを特徴とする請求項４記載の計算機システム。

【請求項６】　複数のサーバに存在するディレクトリが異なる同一ファイルのファイル名を所定のディレクトリのファイル名に一致化させるパス一致化手段をクライアント計算機に設けたことを特徴とする請求項１から請求項５のうちのいずれか１項記載の計算機システム。

【請求項７】　第２のキャッシュ中にキャッシュされているプログラムファイルを外部から取り込み、この外部から取り込んだプログラムファイルを前記第２のキャッシュ中のプログラムファイルと比較し、更新されているか否かをチェックする更新チェック手段をクライアント計算機に設けたことを特徴とする請求項１から請求項６のうちのいずれか１項記載の計算機システム。

【請求項８】　第２のキャッシュ中にキャッシュされているプログラムファイルを外部から取り込み、この外部から取り込んだプログラムファイルを前記第２のキャッシュ中のプログラムファイルと比較し、更新されているか否かをチェックし、更新されていた場合、さらにそのファイルの依存関係解析を行なってロードする更新チェック・ロード手段をクライアント計算機に設けたことを

2

特徴とする請求項１から請求項６のうちのいずれか１項記載の計算機システム。

【請求項９】　プログラムに使用期限がある場合に、ユーザの指定した使用期限と実際の使用頻度にあわせて使用期限が近付いていることの警告を行う使用期限チェック手段をクライアント計算機に設けたことを特徴とする請求項１から請求項８のうちのいずれか１項記載の計算機システム。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】この発明は計算機システムに係り、特にネットワークからプログラムをダウンロードして実行するのに適した計算機システムに関するものである。

【０００２】

【従来の技術】図２２は従来の計算機システムを組み込んだネットワークの構成図であり、図において、２１はプログラムを実行するクライアント計算機、２２はプログラムをクライアント計算機２１に送り出すサーバ計算機、２３はクライアント計算機２１とサーバ計算機２２とを接続するネットワーク、２４はクライアント計算機２１に設けられた、プログラムをロードして実行するプログラムロード・実行手段、２５はサーバ計算機２２に設けられた、プログラムを送出するプログラム送出手段、２６はプログラムロード・実行手段２４に含まれ、キャッシュ２８を制御するキャッシュ制御手段、２７はクライアント計算機２１に接続され、キャッシュ２７を含む不揮発性記憶装置、２８は不揮発性記憶装置２７上のキャッシュ、２９はサーバ計算機２２に接続されている不揮発性記憶装置、３０は不揮発性記憶装置２９上に保存されているプログラムである。クライアント計算機２１と不揮発性記憶装置２７とは計算機システムを構成する。

【０００３】次に動作について説明する。クライアント計算機２１は、ハイパーテキスト言語（Ｈｙｐｅｒ　ＴｅｘｔＭａｒｋｕｐ　Ｌａｎｇｕａｇｅ（以下、「ＨＴＭＬ」と略記する））を使用したWWW（Ｗｏｒｌｄ　Ｗｉｄｅ　Ｗｅｂ）等のネットワーク２３上で、プログラムロード・実行手段２４により、Ｊａｖａ言語などの言語で記述された実行可能なプログラムをサーバ計算機２２からダウンロードして実行する。

【０００４】すなわち、まず、プログラムロード・実行手段２４は、ネットワーク２３を介して、サーバ計算機２２のプログラム送出手段２５にアクセスしてプログラム送出手段２５内に保持されているＨＴＭＬで記載されたＨＴＭＬ文書を閲覧する。閲覧したＨＴＭＬ文書は不揮発性記憶装置２７内のキャッシュ２８に保存される（後に読み出すためにキャッシュ中に一時的にプログラム、データ等を記憶する動作を以後「キャッシュする」と表現する）。このとき閲覧したＨＴＭＬ文書にプログ

ラム３０の実行が指示されていれば、プログラムロード・実行手段２４はプログラム送出手段２５にプログラム３０の送出を依頼して、プログラム３０をダウンロードし、実行する。通常、プログラム３０もキャッシュ２８中にキャッシュされ、次回にアクセスするときにはこのキャッシュ２８中に保持されているプログラム３０が利用される。プログラム３０がサーバ計算機２２により更新されたときに、更新されたプログラム３０をダウンロードしたい場合には、クライアント計算機２１はプログラムロード・実行手段２４に再読み込みを指示する。このようにすることにより、ユーザはＨＴＭＬ文書を閲覧するのに、クライアント計算機２１上にプログラム３０を保存せずに必要なときのみサーバ計算機２２からダウンロードすればよく、サーバ計算機２２によるプログラム３０の一元管理が可能となる。

【０００５】

【発明が解決しようとする課題】従来の計算機システムは以上のように構成されているので、キャッシュ２８がＨＴＭＬ文書やプログラムなどの他のデータにより溢れ、プログラム３０の内容がキャッシュ２８から消去されてしまうことがある。このような場合、再びプログラムロード・実行手段２４がプログラム３０を利用しようとすると、サーバ計算機２２のプログラム送出手段２５にプログラム３０の送出を依頼する必要がある。このことはクライアント計算機２１とサーバ計算機２２とがネットワーク２３により常に接続されている場合や、必要な時に接続可能であり、かつクライアント計算機２１とサーバ計算機２２とがプログラム３０をダウンロードするのに十分な性能を持っている場合には問題ないが、携帯用のクライアント計算機２１を外部に持ち出している場合などクライアント計算機２１とサーバ計算機２２との接続が困難でクライアント計算機２１側の性能も十分でないような場合では、プログラム３０を利用できないという課題があった。また、プログラムファイルがＣＤ－ＲＯＭ等に記録されている場合にも、ＣＤ－ＲＯＭを携帯しないとそのプログラムを実行できないという課題があった。

【０００６】そのような携帯用のクライアント計算機２１が不揮発性記憶装置２７上にキャッシュ２８のみでなく、ユーザが操作可能なファイルシステムを備えている場合には、プログラム３０をそのファイルシステム上に保存しておき、ネットワーク２３にクライアント計算機２１が接続されていない外部でも実行することが可能ではあるが、サーバ計算機２２上のプログラム３０が更新された場合、ユーザが自ら再びそれをダウンロードし、保存しなければならず、サーバ計算機２２によるプログラム３０の一元管理の可能性などの利点を損なってしまうという課題もある。

【０００７】また、プログラム３０が複数のファイルから構成されている場合、プログラムロード・実行手段２

４はプログラム３０中に記録されている情報をもとに自動的に必要なファイルの更なるロード・実行を行うが、常にユーザにとって必要なファイルが正確にロード・実行されるとは限らず、ユーザが、ファイル間の支配・被支配関係である依存関係を判断して複数のファイルをロード・保存する必要が出てくる場合もあるという課題もあった。

【０００８】この発明は上記のような課題を解決するためになされたもので、携帯用のクライアント計算機などネットワーク接続が常には行なえないようなクライアント計算機端末においても、ネットワーク上のプログラムやＣＤ－ＲＯＭ等の記録媒体上のプログラムを常に実行できるようにクライアント計算機に保存でき、ユーザが簡単にプログラムの保存や更新を行うことのできる計算機システムを得ることを目的とする。

【０００９】また、通常のキャッシュが溢れた場合にもプログラムが勝手に削除されることのない計算機システムを得ることを目的とする。

【００１０】さらに、ユーザが依存関係をいちいちチェックしなくとも、依存関係にあるファイルを自動的にキャッシュできる計算機システムを得ることを目的とする。

【００１１】

【課題を解決するための手段】請求項１記載の発明に係る計算機システムは、プログラムファイルを一時的に保存する第１のキャッシュと、不揮発性記憶装置中に設けられ、少なくともユーザの指定するプログラムを保存し、ユーザの消去の指示があるまでそのプログラムを保存し続ける第２のキャッシュとを備えたものである。

【００１２】請求項２記載の発明に係る計算機システムは、第１のキャッシュ中へのプログラムファイルの保存の事実と、保存されたプログラムファイルの依存関係とを記録するキャッシュ履歴簿を不揮発性記憶装置中に設けたものである。

【００１３】請求項３記載の発明に係る計算機システムは、プログラムファイルの依存関係解析機能を持ち、プログラムをロード、キャッシュするプログラムロード・キャッシュ手段をクライアント計算機に設けたものである。

【００１４】請求項４記載の発明に係る計算機システムは、ユーザから削除するように指示のあったファイル群を第２のキャッシュから削除し、第１のキャッシュへコピーするキャッシュ移動手段をクライアント計算機に設けたものである。

【００１５】請求項５記載の発明に係る計算機システムは、第２のキャッシュ中のファイル群の削除履歴簿を不揮発性記憶装置中に設けたものである。

【００１６】請求項６記載の発明に係る計算機システムは、複数のサーバに存在するディレクトリが異なる同一ファイルのファイル名を所定のディレクトリのファイル

名に一致化させるパス一致化手段をクライアント計算機に設けたものである。

【００１７】請求項７記載の発明に係る計算機システムは、第２のキャッシュ中にキャッシュされているプログラムファイルを外部から取り込み、この外部から取り込んだプログラムファイルを第２のキャッシュ中のプログラムファイルと比較し、更新されているか否かをチェックする更新チェック手段をクライアント計算機に設けたものである。

【００１８】請求項８記載の発明に係る計算機システムは、第２のキャッシュ中にキャッシュされているプログラムファイルを外部から取り込み、この外部から取り込んだプログラムファイルを第２のキャッシュ中のプログラムファイルと比較し、更新されているか否かをチェックし、更新されていた場合、さらにそのファイルの依存関係解析を行なってロードする更新チェック・ロード手段をクライアント計算機に設けたものである。

【００１９】請求項９記載の発明に係る計算機システムは、プログラムに使用期限がある場合に、ユーザの指定した使用期限と実際の使用頻度にあわせて使用期限が近付いていることの警告を行う使用期限チェック手段をクライアント計算機に設けたものである。

【００２０】

【発明の実施の形態】以下、この発明の実施の一形態を説明する。

実施の形態１．図１はこの発明の実施の形態１の計算機システムを組み込んだネットワークを示す構成図である。図において、１はプログラムを実行するクライアント計算機、２はプログラムをクライアント計算機１に送り出すサーバ計算機、３はクライアント計算機１とサーバ計算機２とを接続するネットワーク、４はクライアント計算機１に設けられた、プログラムをロードして実行するプログラムロード・実行手段、５はサーバ計算機２に設けられた、プログラムを送出するプログラム送出手段、６はプログラムロード・実行手段４に含まれ、キャッシュ８，１１を制御するキャッシュ制御手段、７はクライアント計算機１に接続され、キャッシュ８，１１を含む不揮発性記憶装置、８は不揮発性記憶装置７上の従来から用いられている通常キャッシュ（第１のキャッシュ）、９はサーバ計算機２に接続されている不揮発性記憶装置、１０は不揮発性記憶装置９上に保存されているプログラム、１１はユーザの消去の指示があるまでデータを保存し続けることのできる永続キャッシュ（第２のキャッシュ）である。クライアント計算機１と不揮発性記憶装置７とは、この発明の計算機システムを構成する。

【００２１】図２はプログラムロード・実行手段４の動作を示すフローチャートである。以下、図２のフローチャートを参照しながらプログラムロード・実行手段４の動作を説明する。

【００２２】ユーザが永続キャッシュ１１へ保存するプログラムを指定し、その保存の実行開始命令を計算機システムに入力すると、プログラムロード・実行手段４は、まず、そのプログラムファイルが永続キャッシュ１１にキャッシュされているか、キャッシュ制御手段６を介して問い合わせる（ステップＳＴ１）。キャッシュされている場合、キャッシュ制御手段６を介してそのプログラムファイルをロードする（ステップＳＴ２）。キャッシュされていない場合、再びキャッシュ制御手段６を介してそのプログラムファイルが通常キャッシュ８にキャッシュされているか否かを問い合わせる（ステップＳＴ３）。

【００２３】そのプログラムファイルが通常キャッシュ８にキャッシュされている場合、通常キャッシュ８よりプログラムファイルをロードし（ステップＳＴ５）、キャッシュされていない場合、ネットワーク３を介してプログラム送出手段５よりプログラムファイルをダウンロードする（ステップＳＴ４）。

【００２４】通常キャッシュ８またはプログラム送出手段５よりプログラムファイルがロードされると、それがユーザが永続キャッシュに保存するように指定したプログラムファイル、またはユーザが保存するように指定したプログラムファイルが必要としたためロードすることとなったファイルであるか否かをチェックする（ステップＳＴ６）。保存指定されたプログラムファイルまたは保存指定されたプログラムファイルが必要としたファイルである場合、キャッシュ制御手段６を介し永続キャッシュ１１に保存し（ステップＳＴ７）、保存指定されたプログラムファイルまたは保存指定されたプログラムファイルが必要としたファイル以外のファイルである場合には、通常キャッシュ８にキャッシュする（ステップＳＴ８）。通常キャッシュ８にキャッシュする場合にはキャッシュ制御手段６のキャッシュ制御方針に従ってキャッシュ溢れ対策が行なわれる。例えば、最も使われておらず、最も古くキャッシュされたファイルを通常キャッシュ８から削除した上で新たなファイルをキャッシュする。

【００２５】ファイルのロードおよびキャッシュ処理が終了すると、プログラムを実行すると共に、その実行の際にそのプログラムファイルが必要とするファイルの相互の支配・被支配関係を解析する依存関係解析を行う（ステップＳＴ９）。依存関係解析および実行に伴って動的に必要とされロードする必要のあるファイルが存在しない場合（ステップＳＴ１０）、キャッシュ処理は終了し、そうでない場合には、キャッシュのチェックおよびロード（ステップＳＴ１）に戻る。

【００２６】以上のように、この実施の形態１によれば、プログラムファイルをプログラム送出手段５よりダウンロードすることができない場合、すなわちクライアント計算機１が携帯用である等の場合には、通常キャッ

シュ８の中にプログラムがキャッシュされておらず通常キャッシュ８からロードできなくとも、ユーザの指定したプログラムファイル及び指定したプログラムファイルが必要としたファイルについては永続キャッシュ１１中に保存されているため、永続キャッシュ１１からそのプログラムファイルをロードすることによりプログラムを実行することができる効果が得られる。また永続キャッシュ１１にプログラムを保存するにあたって、ユーザが保存指定したプログラムファイルまたはそのプログラムファイルが必要とするファイルであるか否かを判定するため、永続キャッシュ１１には本当に保存しておきたいプログラムファイルのみ保存され、キャッシュ領域を無駄に消費しないという効果が得られる。さらにプログラムがモジュール化されており数多くのファイルにより構成される場合でも、プログラムロード・実行手段の依存関係解析機能によりユーザがファイル名を詳細に指定しなくとも、必要なプログラムファイル群を保存できる効果が得られる。なお、永続キャッシュ１１中のプログラムファイル群を更新したい場合には、保存処理と同様の処理を行う。

【００２７】実施の形態２．図３はこの発明の実施の形態２の計算機システムを組み込んだネットワークを示す構成図である。図３において、図１の実施の形態１の構成要素と同一の構成要素には同一の番号を付し、その説明を省略する。

【００２８】図３において、１ａはクライアント計算機、７ａは不揮発性記憶装置、１２はプログラムファイルを通常キャッシュ８にキャッシュする場合に依存関係解析の結果を記録するキャッシュ履歴簿、１３は通常キャッシュ８から永続キャッシュ１１にプログラムファイルをコピーするキャッシュコピー手段である。図４はキャッシュ履歴簿１２の記憶内容を示す図であり、ファイル名と依存関係とを記録してある。図４の例では、ユーザの指示によりファイル名ｍａｉｎ．ｃｌａｓｓのファイルが記録され、ファイルｍａｉｎ．ｃｌａｓｓを実行するのに必要なファイルとしてファイル名ｓｕｂ．ｃｌａｓｓのファイルがファイルｍａｉｎ．ｃｌａｓｓにより指示されたことを表している。クライアント計算機１ａと不揮発性記憶装置７ａとはこの発明の計算機システムを構成する。

【００２９】次に動作について説明する。図５はこの実施の形態２のプログラムロード・実行手段４の動作を示すフローチャートであり、図６はキャッシュコピー手段１３の動作を示すフローチャートである。図５において、図２に示す実施の形態１のプログラムロード・実行手段４の動作ステップと同一の動作を行う動作ステップには同一の番号を付し、その説明を省略する。

【００３０】この実施の形態２においては、ステップＳＴ９でプログラムを実行して依存関係解析を行った後、保存指定されたプログラムファイル及びそのプログラム

ファイルが必要とするファイル以外のファイルを通常キャッシュ８にキャッシュした（ステップＳＴ８）場合、ステップＳＴ９における依存関係解析の結果をキャッシュ履歴簿１２に記録する（ステップＳＴ１１）。すなわち、永続キャッシュ１１に保存すべきファイルを指定していない場合には、永続キャッシュ１１への保存は行なわれない。また、複数のファイルから構成されるプログラムファイル群においては、永続キャッシュ１１への保存を途中から指定すれば、指定したファイル以降のファイルについては永続キャッシュへ１１の保存が可能になるが、それではプログラムの開始地点となるプログラムファイルが保存されていないことになり、意味をなさない。このような場合に、通常キャッシュ８の大きさやキャッシュ制御手段６の制御方針にもよるが、永続キャッシュ１１に保存されなかったプログラムファイルも通常キャッシュ８にはキャッシュされている可能性が高い。そこで、通常キャッシュ８へのキャッシュ動作の依存関係解析結果をキャッシュ履歴簿１２に記録し、通常キャッシュ８から永続キャッシュ１１へのプログラムファイルのコピーに利用するのである。

【００３１】このキャッシュ履歴簿１２には、図４に示すように、プログラムファイルのファイル名と共に、ステップＳＴ９のプログラム実行、依存関係解析の結果が付記されている。したがって、プログラム実行途中に永続キャッシュ１１への保存を指定した場合、このキャッシュ履歴簿１２に記録されている依存関係を逆にたどればプログラムの開始地点を特定でき、この開始地点のプログラムファイルから始まり、順にキャッシュ履歴簿１２をたどってプログラムファイルのロード、永続キャッシュ１１への保存を行なえば、保存を指示した地点までのファイルが保存され、また、それ以降に必要となったファイルは実施の形態１と同様にして保存できる。また、プログラム実行中であるため、永続キャッシュ１１に保存されなかったプログラムファイルも通常キャッシュ８にキャッシュされている可能性は非常に高いため、通常キャッシュ８から永続キャッシュ１１へ移動させれば、ネットワークを使うオーバヘッドが減り、永続キャッシュ１１や通常キャッシュ８の有効的な利用となる。さらに、通常キャッシュ８にプログラムが残っていれば、永続キャッシュ１１への移動が可能となるため、一旦ネットワークからの接続を切ってしまった場合でも永続キャッシュ１１への保存が可能となる。

【００３２】次に、キャッシュコピー手段１３による通常キャッシュ８から永続キャッシュ１１へのプログラムファイルのコピー動作を図６のフローチャートを参照しながら説明する。ユーザが、あるプログラムファイルを通常キャッシュ８から永続キャッシュ１１へコピーすることを指定すると、キャッシュコピー手段１３はキャッシュする先頭のファイルを探索する。すなわち、まず、最初のコピーターゲットファイルをユーザの指定したフ

ァイルとする（ステップＳＴ２１）。

【００３３】次に、キャッシュ履歴簿１２を検索し、コピーターゲットファイルの読み込みを指示した履歴の記録があるか否かを判定する（ステップＳＴ２２）。検索の結果コピーターゲットファイルの読み込みを指示した履歴の記録がキャッシュ履歴簿１２中に存在しない場合には、キャッシュコピー手段１３はプログラムファイルのコピーを諦めて動作を終了する。読み込みを指示した履歴の記録が存在する場合には、この読み込みを指示したファイルを新しいコピーターゲットファイルとする（ステップＳＴ２３）。

【００３４】次に、この新しいコピーターゲットファイルが依存関係の先頭にあるか否かを判断し（ステップＳＴ２４）、先頭になければ、ステップＳＴ２２～ＳＴ２４の動作を繰り返し、依存関係の先頭にあるコピーターゲットファイル（すなわち、ユーザの指示したファイル（図４の例ではファイルｍａｉｎ．ｃｌａｓｓ））を求める。

【００３５】依存関係の先頭にあるファイルが求められたら、このファイルを先頭として通常キャッシュ８から永続キャッシュ１１にコピーする（ステップＳＴ２５）。

【００３６】以上のように、この実施の形態２によれば、永続キャッシュ１１に保存し損なったプログラムも、ネットワーク３を介さずに通常キャッシュ８からコピーでき、ネットワーク３を用いるオーバヘッドを減らすことができるという効果が得られる。

【００３７】実施の形態３．図７はこの発明の実施の形態３の計算機システムを組み込んだネットワークを示す構成図である。図７において、図１の実施の形態１の構成要素と同一の構成要素には同一の番号を付し、その説明を省略する。

【００３８】図７において、１ｂはクライアント計算機、１４はプログラムロード・実行手段４と同様なプログラム依存関係解析機能を持ち、プログラムをロード・キャッシュするプログラムロード・キャッシュ手段である。クライアント計算機１ｂ及び不揮発性記憶装置７はこの発明の計算機システムを構成する。

【００３９】実施の形態１の計算機システムにおいては、プログラムが複数のファイルで構成されている場合には、２番目以降のファイルはプログラムを実行した後に永続キャッシュ１１にキャッシュされることとなる（図２のステップＳＴ６，ＳＴ９，ＳＴ１０参照）。これは、実施の形態１においては、プログラムロード・実行手段４は実行時のロード機能のみ提供していれば十分であるからであるが、プログラムを実行しないでもプログラムファイル群を永続キャッシュ１１に保存しておきたい場合もある。この実施の形態３のプログラムロード・キャッシュ手段１４はこのような要望に応えるものである。

【００４０】次に動作について、図８のプログラムロード・キャッシュ手段１４の動作を示すフローチャートを参照しながら説明する。プログラムロード・キャッシュ手段１４はユーザより保存するプログラムファイルを指定されると、プログラム送出手段５に依頼してそのファイルをネットワーク３を介してダウンロードする（ステップＳＴ３１）。

【００４１】次に、ダウンロードしたプログラムファイルをキャッシュ制御手段６を介して永続キャッシュ１１にキャッシュする（ステップＳＴ３２）。

【００４２】さらに、キャッシュしたプログラムファイルの依存関係解析を行ない（ステップＳＴ３３）、この依存関係解析の結果更にロードすべきファイルがあるか否かを判定する（ステップＳＴ３４）。この判定の結果、更にロードすべきファイルが存在する場合には、ステップＳＴ３１に戻ってファイルのロードを行う。ロードすべきファイルが存在しない場合には動作を終了する。

【００４３】以上のように、この実施の形態３によれば、ユーザはプログラムを実行することなくプログラムファイル群を保存できるだけではなく、プログラムファイル群の構成によってはすべての機能を実行しなければすべてのファイル群を保存することができなかったという実施の形態１の問題点を克服することができる効果が得られる。

【００４４】実施の形態４．図９はこの発明の実施の形態４の計算機システムを組み込んだネットワークを示す構成図である。図９において、図１の実施の形態１の構成要素と同一の構成要素には同一の番号を付し、その説明を省略する。

【００４５】図９において、１ｃはクライアント計算機、１５はユーザが指定したファイル群を永続キャッシュ１１から通常キャッシュ８へ移動するキャッシュ移動手段である。クライアント計算機１ｃ及び不揮発性記憶装置７はこの発明の計算機システムを構成する。

【００４６】実施の形態１の計算機システムにおいては、永続キャッシュ１１へのプログラムファイルの保存が永続キャッシュ溢れにより不可能となった場合には、キャッシュをあきらめるか、通常キャッシュ８にキャッシュするしかない。そこで実施の形態４では、永続キャッシュ１１が溢れた場合、永続キャッシュ１１から通常キャッシュ８へユーザの指定したプログラムファイルを移動させるキャッシュ移動手段１５を設けたのである。

【００４７】次に動作について説明する。図１０はこの実施の形態４のプログラムロード・実行手段４の動作を示すフローチャートであり、図１１はキャッシュ移動手段１５の動作を示すフローチャートである。図１０において、図２に示す実施の形態１のプログラム・ロード実行手段４の動作ステップと同一の動作を行う動作ステップには同一の番号を付し、その説明を省略する。

【００４８】この実施の形態４においては、ステップＳＴ７でキャッシュ制御手段６を介して永続キャッシュ１１にプログラムファイル及びそのプログラムファイルが必要とするファイルをキャッシュした後、永続キャッシュ１１が溢れてキャッシュに失敗したか否かを判定し（ステップＳＴ１２）、キャッシュに成功していればそのままプログラムを実行して依存関係解析を行う（ステップＳＴ９）。永続キャッシュ１１が溢れてキャッシュに失敗した場合には、キャッシュ移動手段１５のサブルーチンを呼び出して実行した（ステップＳＴ１３）後、プログラムを実行して依存関係解析を行う（ステップＳＴ９）。

【００４９】キャッシュ移動手段１５は、まず、永続キャッシュ１１からどのファイル群を通常キャッシュ８に移動するかをユーザに問い合わせる（ステップＳＴ４１）。これは永続キャッシュ１１の記憶内容を勝手に削除しないという永続キャッシュの特徴から避けて通れないステップである。問い合わせは、永続キャッシュ１１にキャッシュされているすべてのファイルについてではなく、ファイル群の単位で問い合わせれば十分である。ユーザが移動すべきファイル群を指示すると、そのファイル群を永続キャッシュ１１から通常キャッシュ８へ移動する（ステップＳＴ４２）。

【００５０】なお、永続キャッシュ１１に永続キャッシュ１１から削除して通常キャッシュ８に移動したファイルの履歴を記録した削除履歴簿を設けてもよい。この削除履歴簿を参照することにより、永続キャッシュ１１中の削除すべきファイルのヒントをユーザに与えることができ、効率の良い永続キャッシュを実現することが可能となる。

【００５１】以上のように、この実施の形態４によれば、永続キャッシュ１１を効率良く利用でき、さらに通常キャッシュ８へ移動されたファイル群の可用性については保証はされないが、この保存を行なってから携帯用計算機システムを持って外出する場合など、運用方法によっては通常キャッシュ８もうまく使用することにより永続キャッシュ１１から削除されたプログラムも使用することができる効果が得られる。

【００５２】実施の形態５．図１２はこの発明の実施の形態５の計算機システムを組み込んだネットワークを示す構成図である。図１２において、図１の実施の形態１の構成要素と同一の構成要素には同一の番号を付し、その説明を省略する。

【００５３】図１２において、１ｄはクライアント計算機、４ａはプログラムロード・実行手段、６ａはキャッシュ制御手段、１６は複数のサーバに存在するディレクトリが異なる同一ファイルのファイル名を所定のディレクトリのファイル名に一致化させるパス一致化手段である。

【００５４】図１３に示すように、計算機ＨｏｓｔＡにはディレクトリ／ｐｒｏｇｒａｍ／ｓｃｈｅｄｕｌｅの下にプログラムを構成する４つのファイルが格納されており、計算機ＨｏｓｔＢにはディレクトリ／ｓｅｒｖｉｃｅ／ｓｃｈｅｄｕｌｅの下に同様に４つのファイルが格納されており、例えば計算機ＨｏｓｔＡは事業所Ａに、計算機ＨｏｓｔＢは事業所Ｂにあり、携帯用計算機システムを持つ営業マンが、ときには事業所Ａを、ときには事業所Ｂを訪れるものとする。実施の形態１の場合、事業所ＡにおいてプログラムファイルＨｏｓｔＡ：／ｐｒｏｇｒａｍ／ｓｃｈｅｄｕｌｅ／ｍａｉｎ．ｃｌａｓｓをプログラム開始ファイルとして永続キャッシュ１１に保存した場合、事業所Ｂで同じプログラム群を更新、保存しようとすればファイルのパスが異なるため事業所Ａで保存したのと同じ内容のファイル群が永続キャッシュ１１中に別に保存されることとなり、永続キャッシュ１１が無駄に消費される。これを避けようとすれば同じプログラムが両方の事業所において提供されているにもかかわらず、営業マンは意識して一方のみを使用しなければならない。そこで、この実施の形態５においては、キャッシュ制御手段６ａにパス一致化手段１６を追加したものである。

【００５５】次に動作について説明する。図１４はこの実施の形態５のプログラムロード・実行手段４ａの動作を示すフローチャートである。図１４において、図２に示す実施の形態１のプログラムロード・実行手段４の動作ステップと同一の動作を行う動作ステップには同一の番号を付し、その説明を省略する。

【００５６】この実施の形態５のプログラムロード・実行手段４ａは、ユーザが永続キャッシュ１１へ保存するプログラムを指定し、その保存の実行開始命令を計算機システムに入力すると、パス一致化手段１６により、ユーザにより指定されたファイル名のパス名（ディレクトリ名）をＢａｓｅ１と変換する（ステップＳＴ１４）。すなわち、パス一致化手段１６は、ユーザが入力したプログラムファイル名の開始ディレクトリが図１３のファイル群１のＢａｓｅとして登録されているか否かを判断し、登録されている場合には入力された開始ディレクトリの如何にかかわらず、ファイル名の開始ディレクトリをＨｏｓｔＡ：／ｐｒｏｇｒａｍ／ｓｃｈｅｄｕｌｅに変換して、以後の動作を行う。以後の動作は実施の形態１のプログラムロード・実行手段４の動作と同一である。

【００５７】以上のように、この実施の形態５によれば、キャッシュ制御手段６ａを介して永続キャッシュ１１にキャッシュする（ステップＳＴ７）か、または永続キャッシュ１１にキャッシュされていてキャッシュ制御手段６ａによってプログラムファイルをロードする場合（ステップＳＴ２）に、図１３のファイル群情報を利用して、プログラムロード・実行手段４ａが欲しているファイルがＨｏｓｔＡ：／ｐｒｏｇｒａｍ／ｓｃｈｅｄｕ

ｌｅ／ｍａｉｎ．ｃｌａｓｓであっても、ＨｏｓｔＢ：／ｐｒｏｇｒａｍ／ｓｃｈｅｄｕｌｅ／ｍａｉｎ．ｃｌａｓｓであっても、ファイル群１中のｍａｉｎ．ｃｌａｓｓを参照する。これにより複数のサーバを同様に扱うことができ、永続キャッシュ１１を無駄にしない効果が得られる。

【００５８】実施の形態６．図１５はこの発明の実施の形態６の計算機システムを組み込んだネットワークを示す構成図である。図１５において、図１の実施の形態１の構成要素と同一の構成要素には同一の番号を付し、その説明を省略する。

【００５９】図１５において、１ｅはクライアント計算機、４ｂはプログラムロード・実行手段、６ｂはキャッシュ制御手段、１７はプログラム送出手段５及び永続キャッシュ１１から同一ファイル名のプログラムファイルをロードして内容を比較し、変更の有無を検出する、キャッシュ制御手段６ｂに設けられた更新チェック手段である。

【００６０】次に動作について説明する。図１６はこの実施の形態６のプログラムロード・実行手段４ｂの動作を示すフローチャートである。図１６において、図２に示す実施の形態１のプログラム・ロード実行手段４の動作ステップと同一の動作を行う動作ステップには同一の番号を付し、その説明を省略する。

【００６１】この実施の形態６においては、永続キャッシュ１１に必要なプログラムファイルが保存されていれば保存されているプログラムファイルをロードし（ステップＳＴ２）、計算機システムがネットワーク３に接続されているか否かを判断する（ステップＳＴ１５）。計算機システムがネットワーク３に接続されていなければロードしたプログラムファイルに基づいてプログラムを実行し、依存関係解析を行う（ステップＳＴ９）。計算機システムがネットワーク３に接続されているときには、更新チェック手段１７がプログラム送出手段５から永続キャッシュ１１に保存されているのと同一のファイル名のプログラムファイルをダウンロードする（ステップＳＴ１６）。

【００６２】次に、更新チェック手段１７はプログラム送出手段５からダウンロードしたプログラムファイルと永続キャッシュ１１中のプログラムファイルとの内容を比較する（ステップＳＴ１７）。その結果内容が相違している場合には変更があった旨を表示してユーザに知らせる（ステップＳＴ１８）。以後、ネットワーク３に接続されていない場合と同様にダウンロードしたプログラムファイルに基づいてプログラムを実行し、依存関係解析を行う（ステップＳＴ９）。

【００６３】この場合、永続キャッシュ１１の内容は勝手に更新しない。更新情報に基づいて更新したい場合には、ユーザはプログラム送出手段５から更新されたプログラムをダウンロードし（ステップＳＴ４）、キャッシュ制御手段６ｂにより永続キャッシュ１１（ステップＳＴ７）または通常キャッシュ８（ステップＳＴ８）にキャッシュすることにより更新する。この際、実施の形態２のように、通常キャッシュ８にキャッシュした場合に依存関係解析結果をキャッシュ履歴簿１２（図３）に記録してもよい（図５のステップＳＴ１１）。

【００６４】以上のように、この実施の形態６によれば、容易に更新チェックができるだけでなく、タイムスタンプなどによる更新チェックではなくプログラムファイルの内容を比較するため、どの機能を司るファイルが更新されているか、大きく更新されているか少なく更新されているかなどの情報をユーザが知ることができる効果が得られる。また、残念ながらプログラムの更新により新たなバグが混入されたり、かえって使い勝手が悪くなってしまうことがよくある。このようなことによる影響を避けるため、永続キャッシュを変更することなく更新されたプログラムの動作チェックなどをすることができる効果が得られる。

【００６５】実施の形態７．図１７はこの発明の実施の形態７の計算機システムを組み込んだネットワークを示す構成図である。図１７において、図１の実施の形態１の構成要素と同一の構成要素には同一の番号を付し、その説明を省略する。

【００６６】図１７において、１ｆはクライアント計算機、４ｃはプログラムロード・実行手段、６ｃはキャッシュ制御手段、１８はプログラム送出手段５及び永続キャッシュ１１から同一ファイル名のプログラムファイルをロードして内容を比較し、変更の有無を検出し、更新されたファイルを永続キャッシュに保存して、更に更新されたファイルの依存関係解析を行って更新されたファイルにより必要とされるようになった新たなファイルをロード・保存する、キャッシュ制御手段６ｂに設けられた更新チェック・ロード手段である。

【００６７】次に動作について説明する。図１８はこの実施の形態７のプログラムロード・実行手段４ｃの動作を示すフローチャートである。図１８において、図１６に示す実施の形態６のプログラム・ロード実行手段４ｂの動作ステップと同一の動作を行う動作ステップには同一の番号を付し、その説明を省略する。

【００６８】この実施の形態７においては、実施の形態６と同様に、永続キャッシュ１１に必要なプログラムファイルが保存されていれば保存されているプログラムファイルをロードし（ステップＳＴ２）、計算機システムがネットワーク３に接続されているか否かを判断する（ステップＳＴ１５）。計算機システムがネットワーク３に接続されていなければロードしたプログラムファイルに基づいてプログラムを実行し、依存関係解析を行う（ステップＳＴ９）。計算機システムがネットワーク３に接続されているときには、更新チェック・ロード手段１８がプログラム送出手段５から永続キャッシュ１１に

保存されているのと同一のファイル名のプログラムファイルをダウンロードする（ステップＳＴ１６）。

【００６９】次に、更新チェック手段１７はプログラム送出手段５からダウンロードしたプログラムファイルと永続キャッシュ１１中のプログラムファイルとの内容を比較する（ステップＳＴ１７）。その結果内容が相違している場合には変更があった旨を表示してユーザに知らせる（ステップＳＴ１８）。

【００７０】この表示によりユーザがプログラムの更新を指示しなければ（ステップＳＴ１９）、実施の形態６と同様に、ダウンロードしたプログラムファイルに基づいてプログラムを実行し、依存関係解析を行う（ステップＳＴ９）。ユーザがプログラムの更新を指示した場合には（ステップＳＴ１９）、更新チェック・ロード手段１８はステップＳＴ１７での比較結果により更新されているプログラムファイルを、実施の形態３のプログラムロード・キャッシュ手段１４と同様に、ロードし（図８のステップＳＴ３１参照）、永続キャッシュ１１にキャッシュし（図８のステップＳＴ３２参照）、そのプログラムファイルの依存解析を行って（図８のステップＳＴ３３参照）プログラムファイルを更新する（図１８のステップＳＴ２０）。

【００７１】以上のように、この実施の形態７によれば、永続キャッシュ１１の更新が容易に、かつ更新をもれなく反映することができる効果が得られる。

【００７２】実施の形態８．図１９はこの発明の実施の形態８の計算機システムを組み込んだネットワークを示す構成図である。図１９において、図１の実施の形態１の構成要素と同一の構成要素には同一の番号を付し、その説明を省略する。

【００７３】図１９において、１ｇはクライアント計算機、１９はプログラムの使用期限をチェックする使用期限チェック手段、２０は使用期限と使用の経歴を記録した使用期限記録簿である。

【００７４】図２０は不揮発性記憶装置７ｂ中に設けられた使用期限記録簿２０の記録内容の一例を示す図である。図２０の例では、ファイル群Ａには使用期限がなく、ファイル群Ｂは使用期限が１９９７年３月１０日までであり、前回１９９７年１月２５日に使用した実績があることを表している。

【００７５】次に動作について説明する。試用あるいは課金のためなどによりプログラムの使用期限がある場合など、永続キャッシュ１１中のプログラムが使用期限切れのため使用できず、永続キャッシュ１１の効果が発揮できない場合がある。この実施の形態８の計算機システムは、例えば週報を作成するためのプログラムのようにユーザは週に一度程度しか使わないなどの情報とユーザが指定する使用期限とを勘案して、使用期限が近付いていることをユーザに警告するものである。

【００７６】図２１はこの実施の形態８のプログラムロード・実行手段４の動作を示すフローチャートである。図２１において、図２に示す実施の形態１のプログラム・ロード実行手段４の動作ステップと同一の動作を行う動作ステップには同一の番号を付し、その説明を省略する。

【００７７】この実施の形態８においては、キャッシュ制御手段６によりプログラムをロードした（ステップＳＴ２）後、次回の使用時期を予測するため、次回使用予想時期＝今回使用時期－前回使用時期＋今回使用時期を演算して求める（ステップＳＴ２１１）。

【００７８】次に、このようにして求めた次回使用予想時期が使用期限を超えている場合には、その旨を表示してユーザに警告する（ステップＳＴ２１２）。

【００７９】また、今回の使用時期が既に使用期限を超えているか否かを判断し（ステップＳＴ２１３）、使用期限を超えてしまっている場合には実行プロセスを終了する。使用期限を超えていない場合には、今回の使用時期を使用期限記録簿２０に記し（ステップＳＴ２１３）、永続キャッシュ１１にキャッシュされているプログラムを実行し、依存関係解析を行う（ステップＳＴ９）。

【００８０】以上のように、この実施の形態８によれば、例えば月報を作成するためのプログラムならば週報プログラムの場合より使用期限と警告を出すべき時の間がより長くなる可能性が高いなどのプログラム使用頻度に応じた指摘を行うことができ、ユーザがプログラムの更新のためにそのプログラムのある事業所に出かけるなどの行動を取るための指針を与えることができる効果が得られる。

【００８１】なお、以上のいずれの実施の形態においても、計算機システムはネットワーク中で用いる場合について説明したが、本発明の計算機システムはネットワーク中での使用に限定されるものではなく、プログラムファイルがＣＤ－ＲＯＭやフロッピーディスク等の記録媒体に記録されていて、このような記録媒体等の外部のプログラムソースからプログラムファイルを取り込んで使用するような場合についても適用し得るものである。

【００８２】

【発明の効果】以上のように、請求項１記載の発明によれば、不揮発性記憶装置中に永続キャッシュを設けるように構成したので、ネットワークに接続されていなくとも、またプログラムを記録した記録媒体が手元になくとも、プログラムが利用可能であるだけでなく、明示的に永続キャッシュを消去しないかぎりプログラムが利用可能であり、さらにプログラムが複数ファイルから構成される場合でもプログラムロード・実行手段により依存関係が解決されてロードされるため、ユーザはどのファイルが必要か詳細に指定する必要なく永続キャッシュに必要なファイルをキャッシュできる効果がある。また、依存解析により必要なファイルのみが永続キャッシュにキ

ャッシュされ、それ以外のファイルは通常キャッシュにキャッシュされるため、永続キャッシュは効率良く利用される効果がある。

【００８３】請求項２記載の発明によれば、不揮発性記憶装置中にキャッシュ履歴簿を設けるように構成したので、プログラム実行前だけでなくプログラム実行途中や終了後からでも遡ってプログラム実行開始時点を容易に特定でき、そのプログラム実行開始時点からのプログラムファイルの永続キャッシュへのキャッシュが可能となるばかりではなく、通常キャッシュにキャッシュされている内容を永続キャッシュにコピーすることにより、通信量の削減が可能となり、さらにその時点でネットワークに接続されていなくとも永続キャッシュの構築が可能となる効果がある。

【００８４】請求項３記載の発明によれば、クライアント計算機にプログラムロード・キャッシュ手段を設けるように構成したので、プログラムを実行しなくとも永続キャッシュにプログラムファイルをキャッシュでき、さらにプログラムファイルが複数のファイルにより構成されている場合にすべての機能を実行することなく、もれなくプログラム全体を構成するファイルをロード、キャッシュすることができる効果がある。

【００８５】請求項４記載の発明によれば、キャッシュ移動手段をクライアント計算機に設けるように構成したので、永続キャッシュ中のプログラムファイルの削除とそのプログラムファイルの通常キャッシュへのコピーにより、効率よく永続キャッシュを維持することが可能となり、永続キャッシュから削除されたプログラムファイルも通常キャッシュから削除されない限りネットワークに接続することなく使用することができる効果がある。

【００８６】請求項５記載の発明によれば、第２のキャッシュ中のファイル群の削除履歴簿を不揮発性記憶装置中に設けるように構成したので、永続キャッシュの削除履歴簿を参照することにより、削除すべき永続キャッシュのヒントをユーザに与えることができ、効率良く永続キャッシュを維持することが可能となる。

【００８７】請求項６記載の発明によれば、バス一致化手段をクライアント計算機に設けるように構成したので、複数の事業所等があるため複数の同じプログラムを提供するサーバがある場合に、１つのサーバからではなくどのサーバからでもプログラムの永続キャッシュへのロードが１つ分の永続キャッシュの容量で可能となり、かつどのサーバからでも永続キャッシュの更新が可能となる効果がある。

【００８８】請求項７記載の発明によれば、更新チェック手段をクライアント計算機に設けるように構成したので、永続キャッシュの更新情報が容易にかつ効率良く判断でき、またユーザの指示がなければ永続キャッシュは更新されないため、これまで安定して動作しているなどの理由によりユーザの満足しているバージョンのプログラムを永続キャッシュに残したまま、更新された版のプログラムを試使用することができる効果がある。

【００８９】請求項８記載の発明によれば、更新チェック・ロード手段をクライアント計算機に設けるように構成したので、プログラムの更新情報が得られるだけでなく、依存関係解析を行うことにより、プログラムが複数のファイル群により構成されている場合でもプログラム全体をもれなく更新することができる効果がある。

【００９０】請求項９記載の発明によれば、使用期限チェック手段をクライアント計算機に設けるように構成したので、試用あるいは課金のためなど、プログラムの使用期限がある場合、使用頻度に応じて使用期限が近付いていることを警告し、ユーザに新たな使用期限を持つプログラムへの更新を促すことができる効果がある。

【図面の簡単な説明】

【図１】　この発明の実施の形態１の計算機システムを組み込んだネットワークを示す構成図である。

【図２】　プログラムロード・実行手段の動作を示すフローチャートである。

【図３】　この発明の実施の形態２の計算機システムを組み込んだネットワークを示す構成図である。

【図４】　実施の形態２のキャッシュ履歴簿の記憶内容を示す図である。

【図５】　実施の形態２のプログラムロード・実行手段の動作を示すフローチャートである。

【図６】　実施の形態２のキャッシュコピー手段の動作を示すフローチャートである。

【図７】　この発明の実施の形態３の計算機システムを組み込んだネットワークを示す構成図である。

【図８】　実施の形態３のプログラムロード・キャッシュ手段の動作を示すフローチャートである。

【図９】　この発明の実施の形態４の計算機システムを組み込んだネットワークを示す構成図である。

【図１０】　実施の形態４のプログラムロード・実行手段の動作を示すフローチャートである。

【図１１】　実施の形態４のキャッシュ移動手段の動作を示すフローチャートである。

【図１２】　この発明の実施の形態５の計算機システムを組み込んだネットワークを示す構成図である。

【図１３】　各計算機の保存するプログラムファイルの分布を表す図である。

【図１４】　実施の形態５のプログラムロード・実行手段の動作を示すフローチャートである。

【図１５】　この発明の実施の形態６の計算機システムを組み込んだネットワークを示す構成図である。

【図１６】　実施の形態６のプログラムロード・実行手段の動作を示すフローチャートである。

【図１７】　この発明の実施の形態７の計算機システムを組み込んだネットワークを示す構成図である。

【図１８】　実施の形態７のプログラムロード・実行手

19

段の動作を示すフローチャートである。

【図１９】　この発明の実施の形態８の計算機システムを組み込んだネットワークを示す構成図である。

【図２０】　実施の形態８の不揮発性記憶装置中に設けられた使用期限記録簿の記録内容の一例を示す図である。

【図２１】　実施の形態８のプログラムロード・実行手段の動作を示すフローチャートである。

【図２２】　従来の計算機システムを組み込んだネットワークの構成図である。

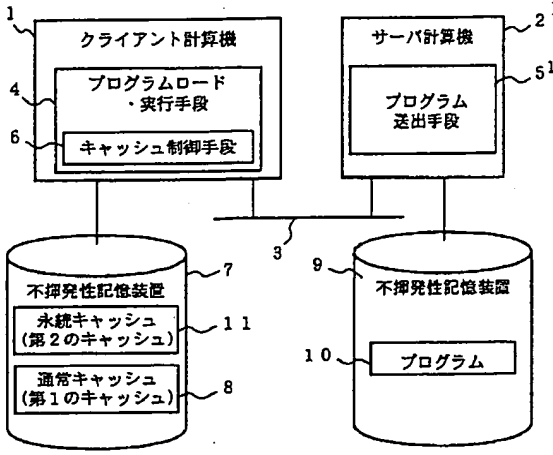
20

【符号の説明】

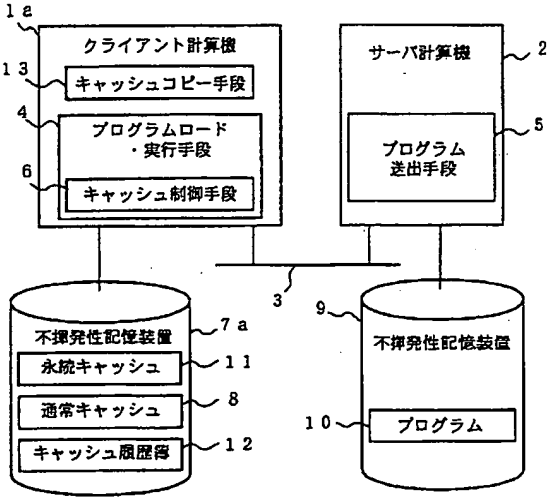
１，１ａ，１ｂ，１ｃ，１ｄ，１ｅ，１ｆ，１ｇ　クライアント計算機、７，７ａ，７ｂ　不揮発性記憶装置、８　通常キャッシュ（第１のキャッシュ）、１０　プログラム、１１　永続キャッシュ（第２のキャッシュ）、１２　キャッシュ履歴簿、１４　プログラムロード・キャッシュ手段、１５　キャッシュ移動手段、１６　パス一致化手段、１７　更新チェック手段、１８　更新チェック・ロード手段、１９　使用期限チェック手段。

10

【図１】



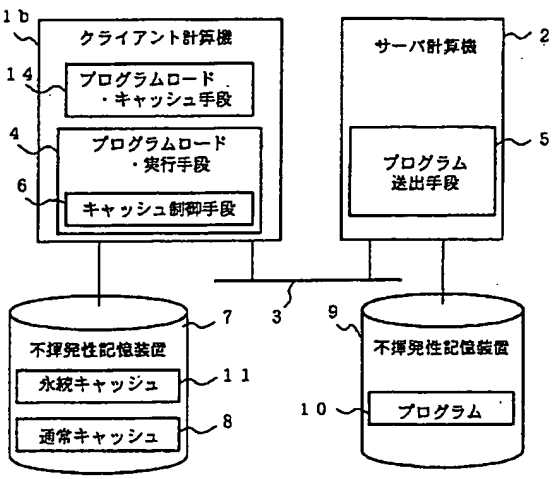
【図３】



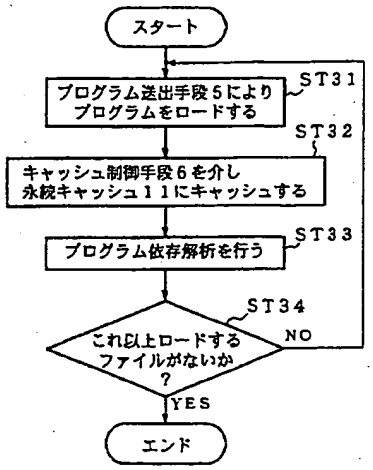
【図４】

ファイル名	依存関係
main.class	ユーザ指示
sub.class	main.class

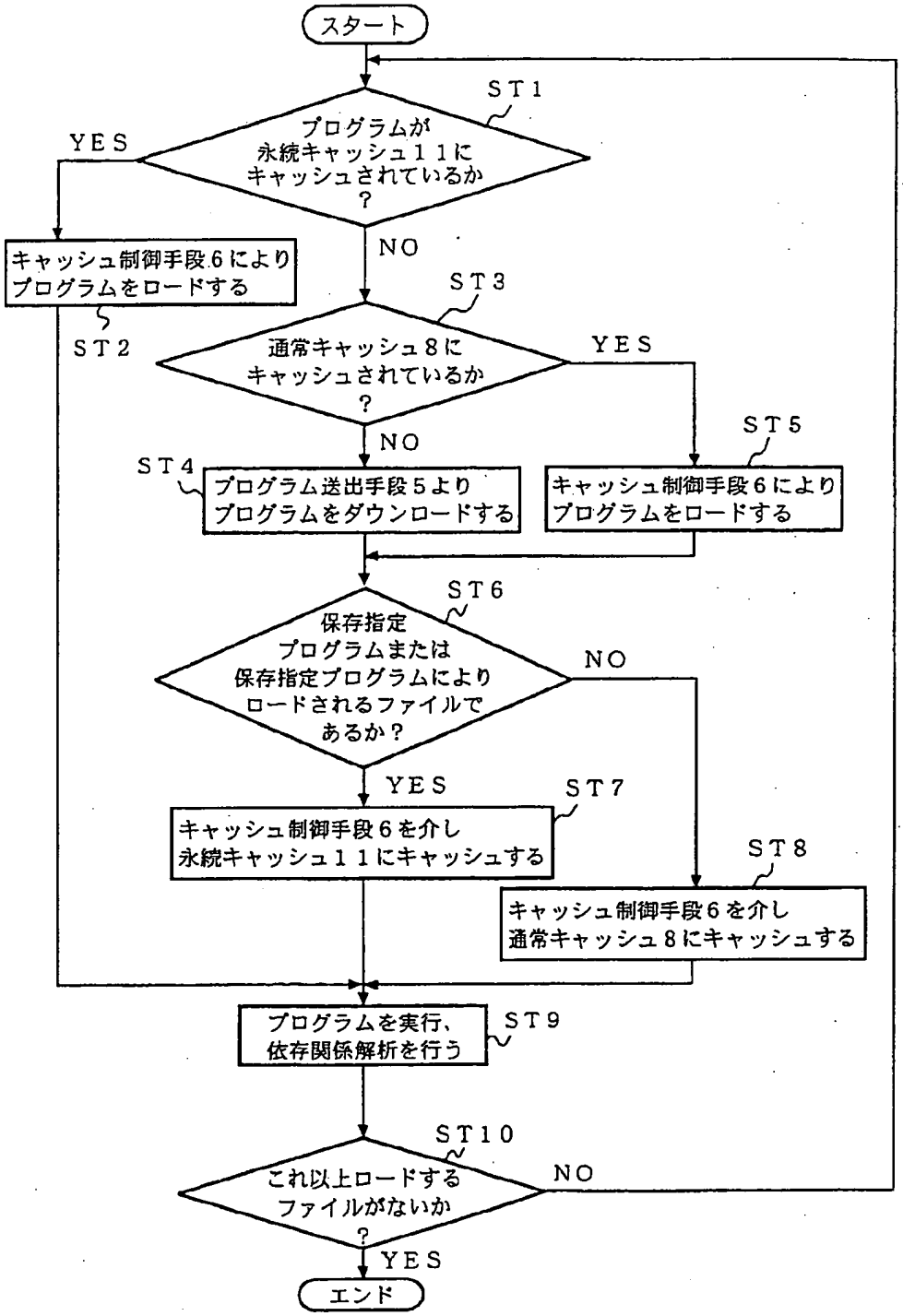
【図７】



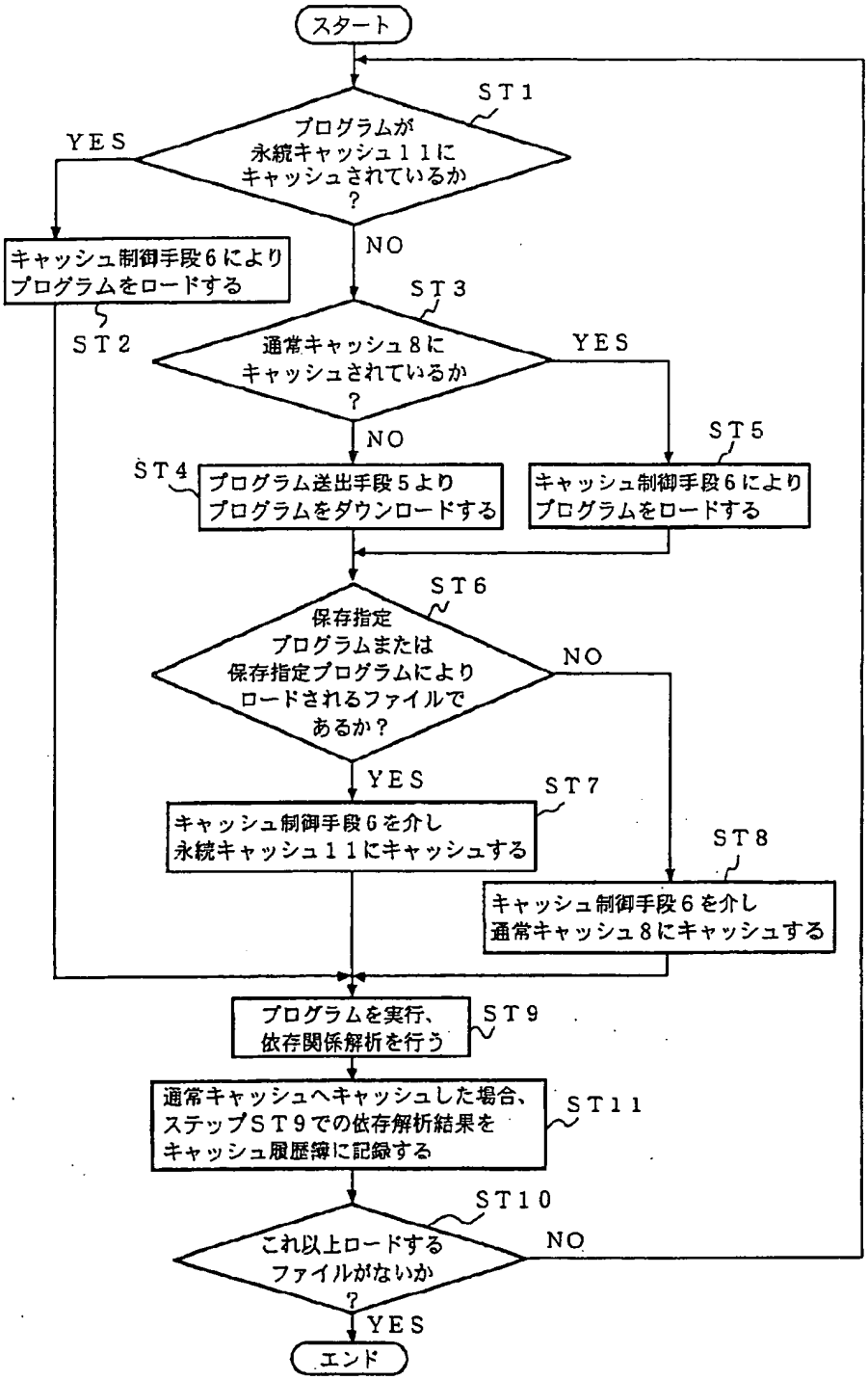
【図８】



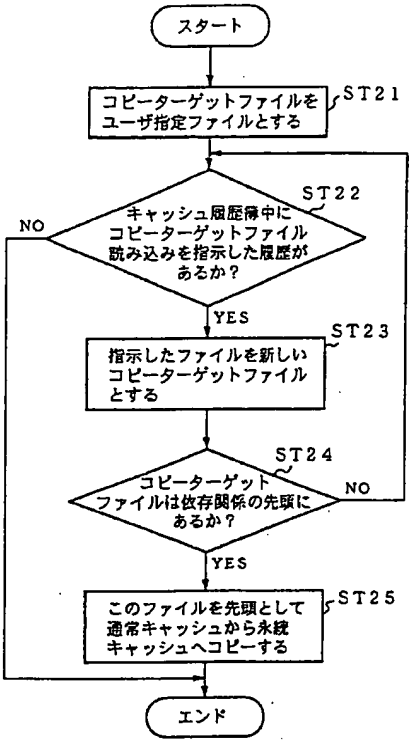
【図２】


スタート
ST1
プログラムが
永続キャッシュ１１に
キャッシュされているか
？
YES
NO
キャッシュ制御手段６により
プログラムをロードする
ST2
ST3
通常キャッシュ８に
キャッシュされているか
？
YES
NO
ST4
プログラム送出手段５より
プログラムをダウンロードする
ST5
キャッシュ制御手段６により
プログラムをロードする
ST6
保存指定
プログラムまたは
保存指定プログラムにより
ロードされるファイルで
あるか？
NO
YES
ST7
キャッシュ制御手段６を介し
永続キャッシュ１１にキャッシュする
ST8
キャッシュ制御手段６を介し
通常キャッシュ８にキャッシュする
プログラムを実行、
依存関係解析を行う
ST9
ST10
これ以上ロードする
ファイルがないか
？
NO
YES
エンド

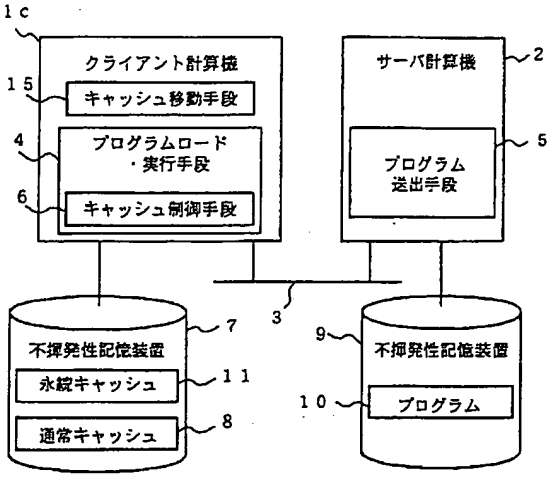
【図５】


スタート
ST1
プログラムが
永続キャッシュ１１に
キャッシュされているか
？
YES
NO
キャッシュ制御手段６により
プログラムをロードする
ST2
ST3
通常キャッシュ８に
キャッシュされているか
？
YES
NO
ST4
プログラム送出手段５より
プログラムをダウンロードする
ST5
キャッシュ制御手段６により
プログラムをロードする
ST6
保存指定
プログラムまたは
保存指定プログラムにより
ロードされるファイルで
あるか？
NO
YES
ST7
キャッシュ制御手段６を介し
永続キャッシュ１１にキャッシュする
ST8
キャッシュ制御手段６を介し
通常キャッシュ８にキャッシュする
プログラムを実行、
依存関係解析を行う
ST9
通常キャッシュへキャッシュした場合、
ステップＳＴ９での依存解析結果を
キャッシュ履歴簿に記録する
ST11
ST10
これ以上ロードする
ファイルがないか
？
NO
YES
エンド

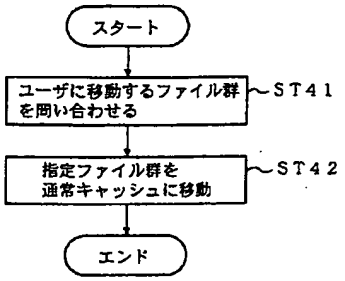
【図６】



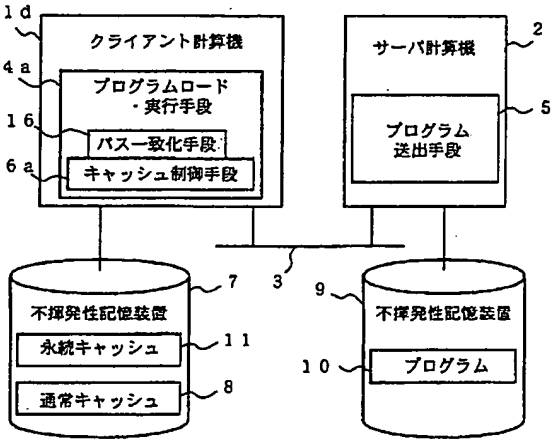
【図９】



【図１１】



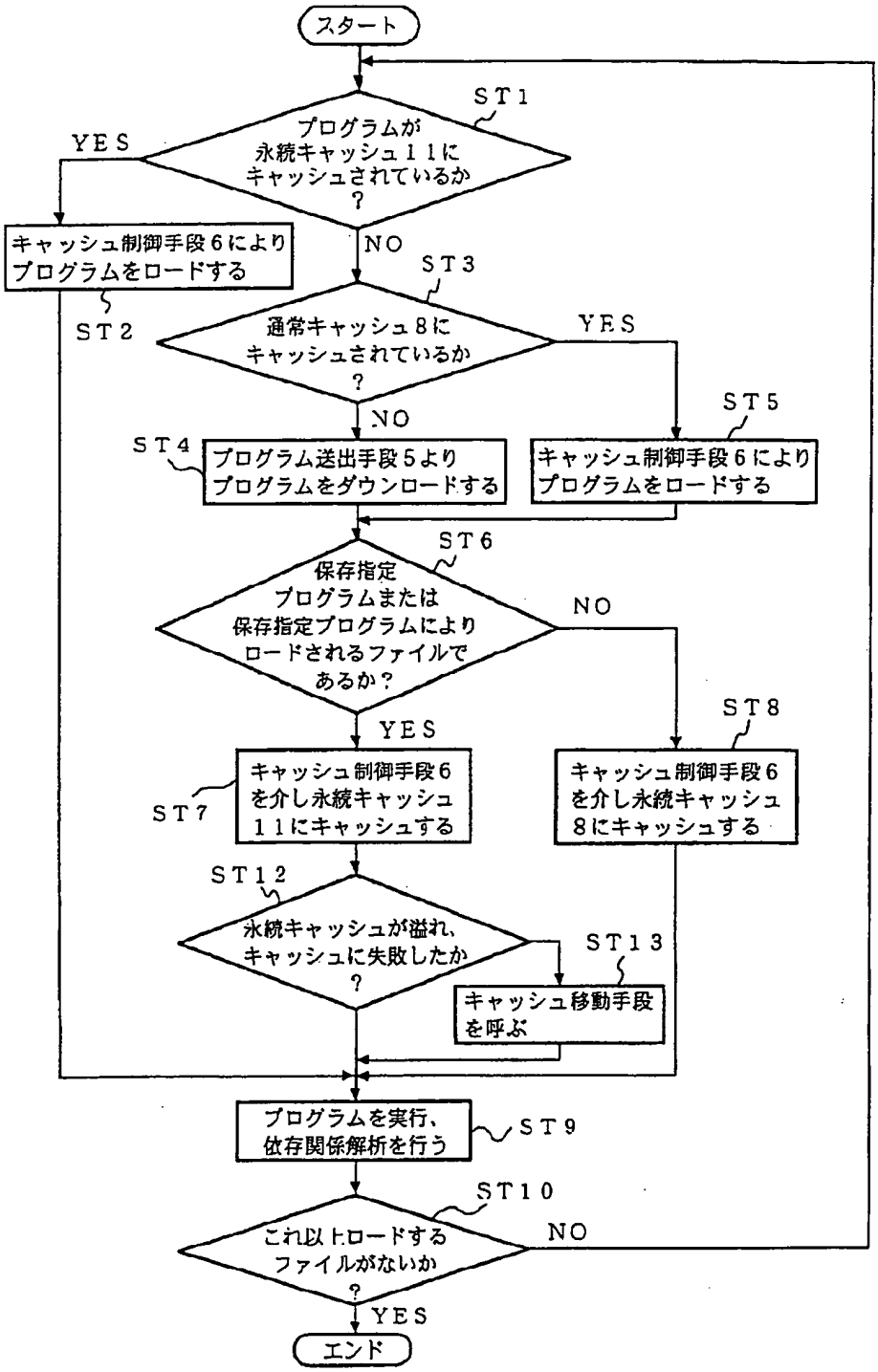
【図１２】



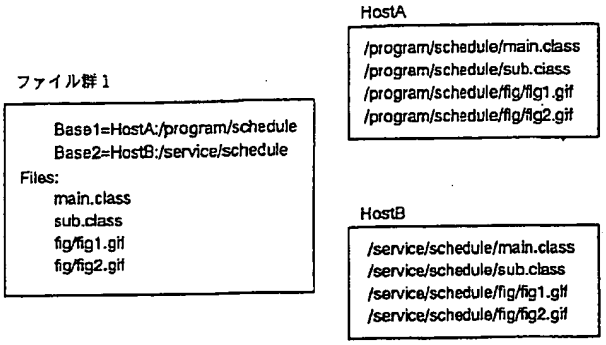
【図２０】

ファイル群	使用期限	前回使用
ファイル群A	なし	----
ファイル群B	97/3/10	97/1/25

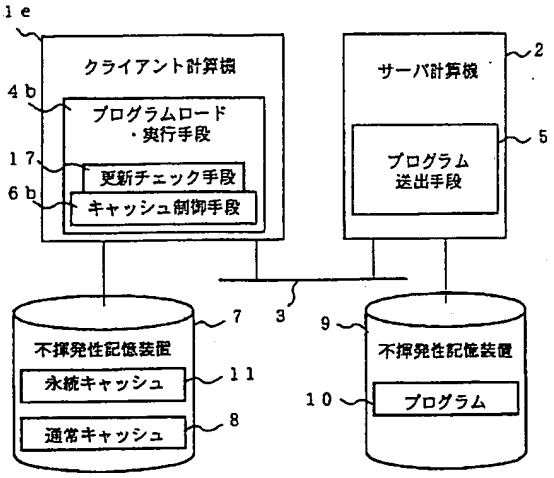
【図１０】


スタート
ST1
プログラムが永続キャッシュ１１にキャッシュされているか？
YES
NO
キャッシュ制御手段６によりプログラムをロードする
ST2
ST3
通常キャッシュ８にキャッシュされているか？
YES
NO
ST4
プログラム送出手段５よりプログラムをダウンロードする
ST5
キャッシュ制御手段６によりプログラムをロードする
ST6
保存指定プログラムまたは保存指定プログラムによりロードされるファイルであるか？
NO
YES
ST7
キャッシュ制御手段６を介し永続キャッシュ１１にキャッシュする
ST8
キャッシュ制御手段６を介し永続キャッシュ８にキャッシュする
ST12
永続キャッシュが溢れ、キャッシュに失敗したか？
ST13
キャッシュ移動手段を呼ぶ
プログラムを実行、依存関係解析を行う
ST9
ST10
これ以上ロードするファイルがないか？
NO
YES
エンド

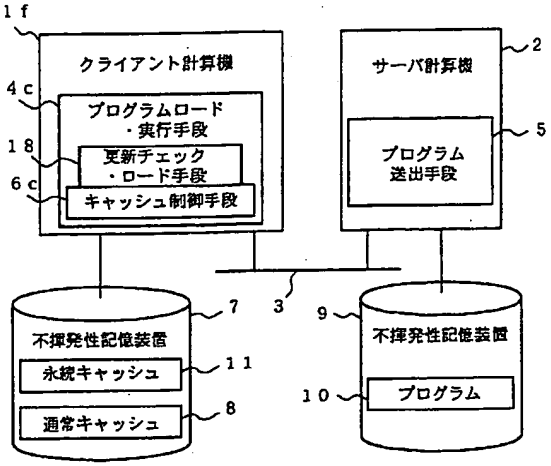
【図１３】



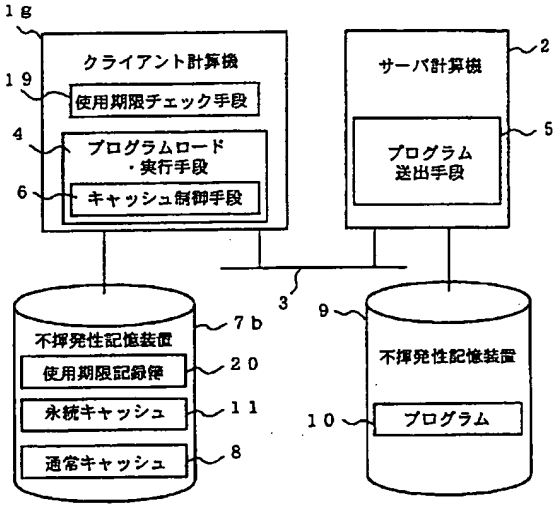
【図１５】



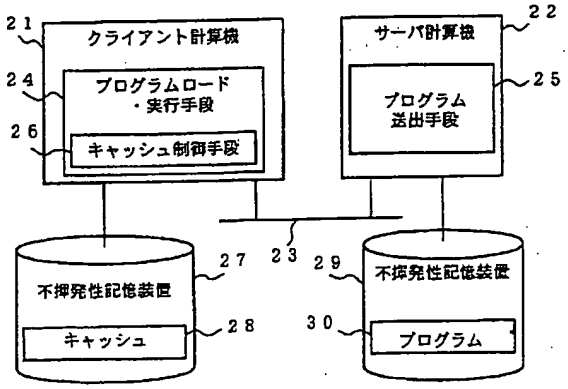
【図１７】



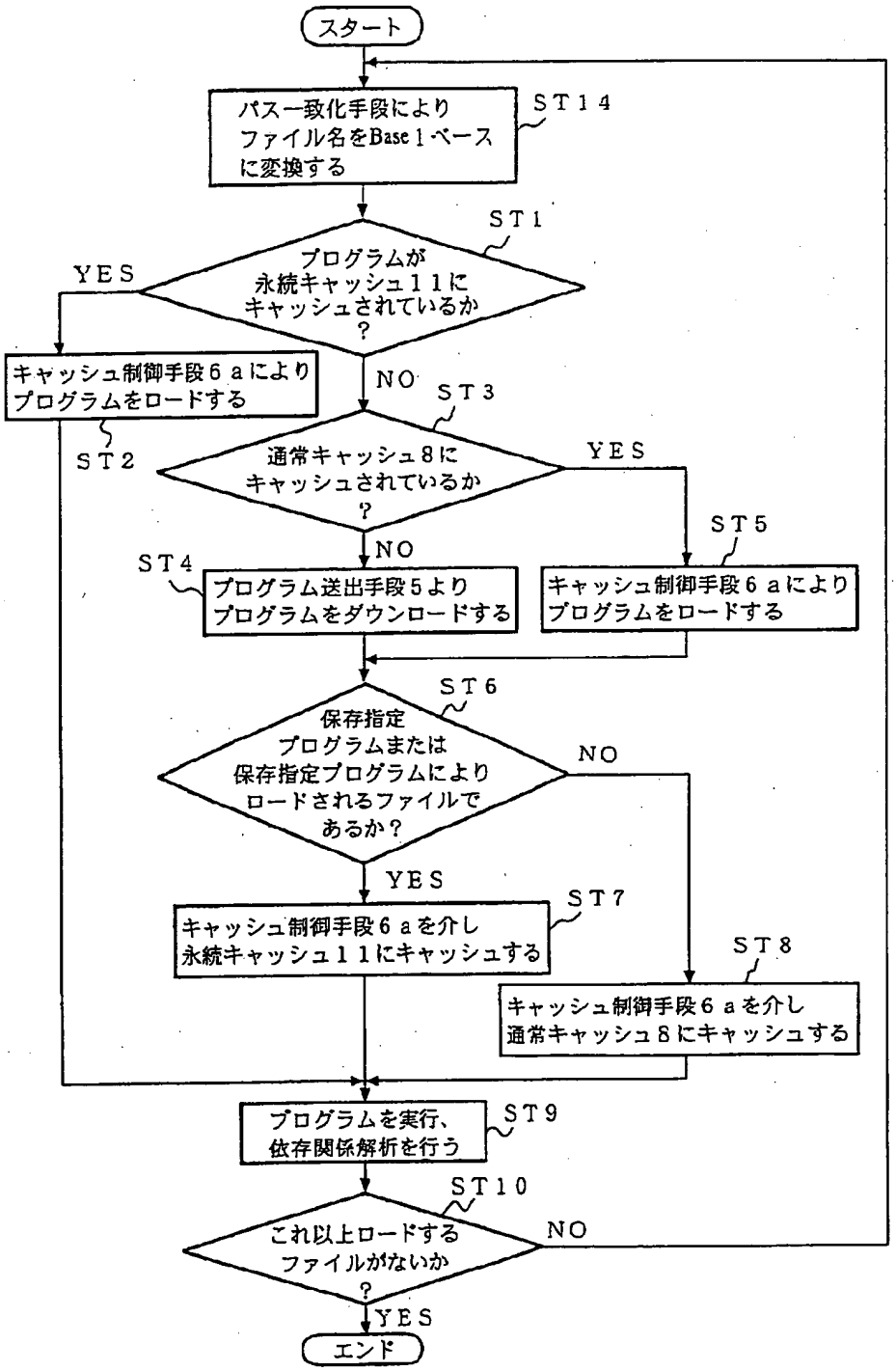
【図１９】



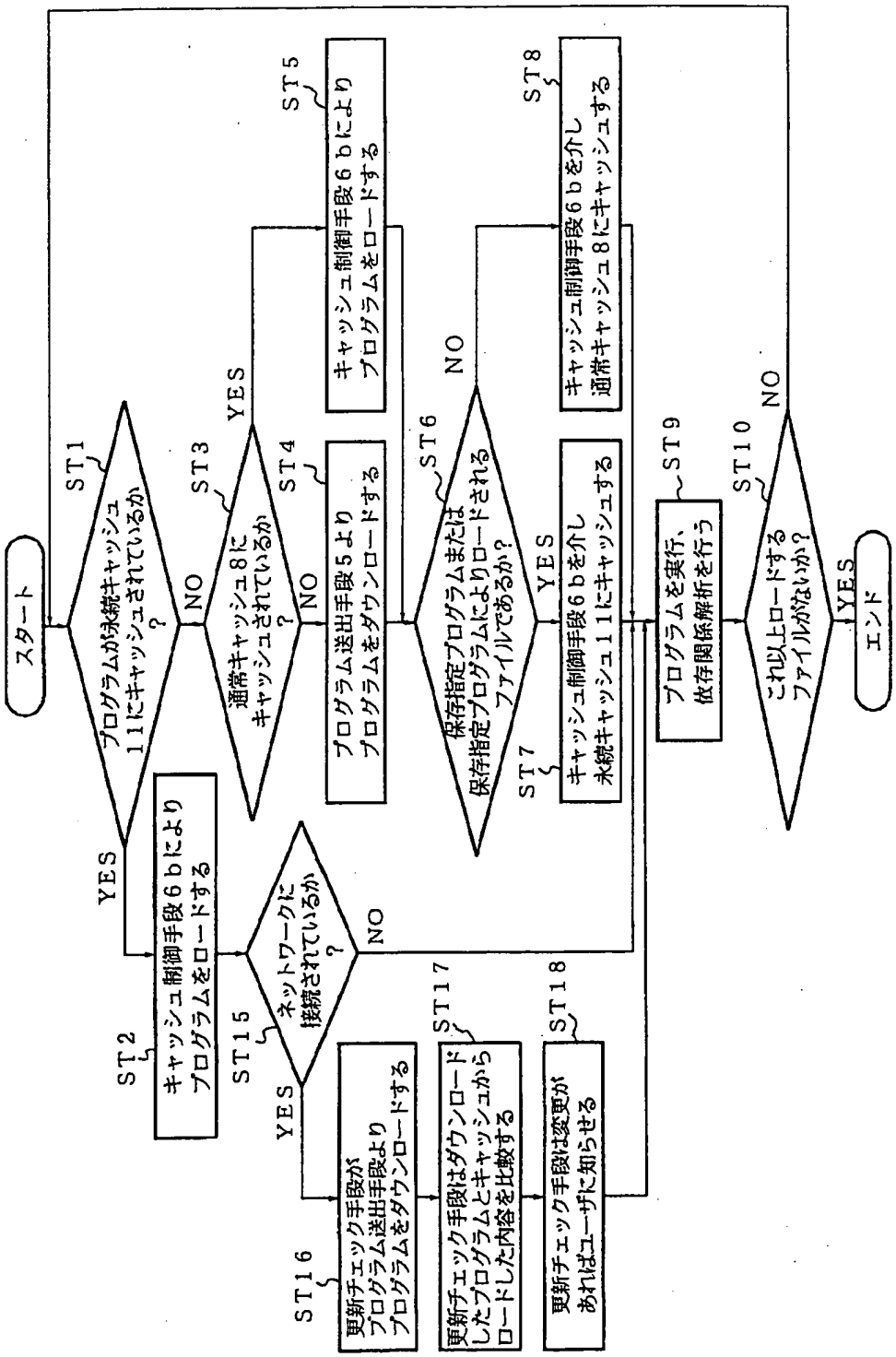
【図２２】



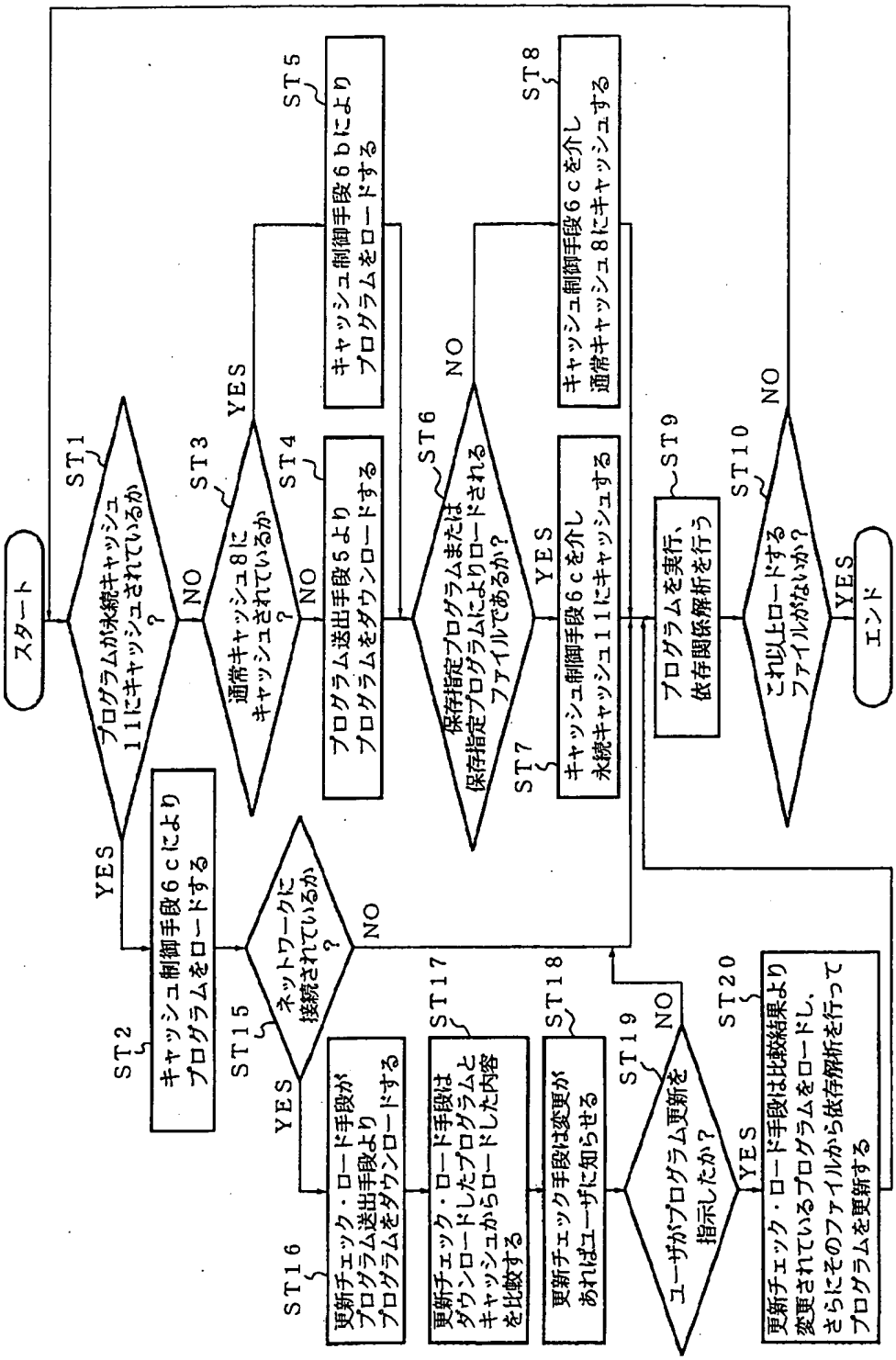
【図１４】


スタート
パス一致化手段により
ファイル名をBase１ベース
に変換する
ST１４
ST１
プログラムが
永続キャッシュ１１に
キャッシュされているか
？
YES
NO
キャッシュ制御手段６ａにより
プログラムをロードする
ST２
ST３
通常キャッシュ８に
キャッシュされているか
？
YES
NO
ST４
プログラム送出手段５より
プログラムをダウンロードする
ST５
キャッシュ制御手段６ａにより
プログラムをロードする
ST６
保存指定
プログラムまたは
保存指定プログラムにより
ロードされるファイルで
あるか？
NO
YES
キャッシュ制御手段６ａを介し
永続キャッシュ１１にキャッシュする
ST７
ST８
キャッシュ制御手段６ａを介し
通常キャッシュ８にキャッシュする
プログラムを実行、
依存関係解析を行う
ST９
ST１０
これ以上ロードする
ファイルがないか
？
NO
YES
エンド

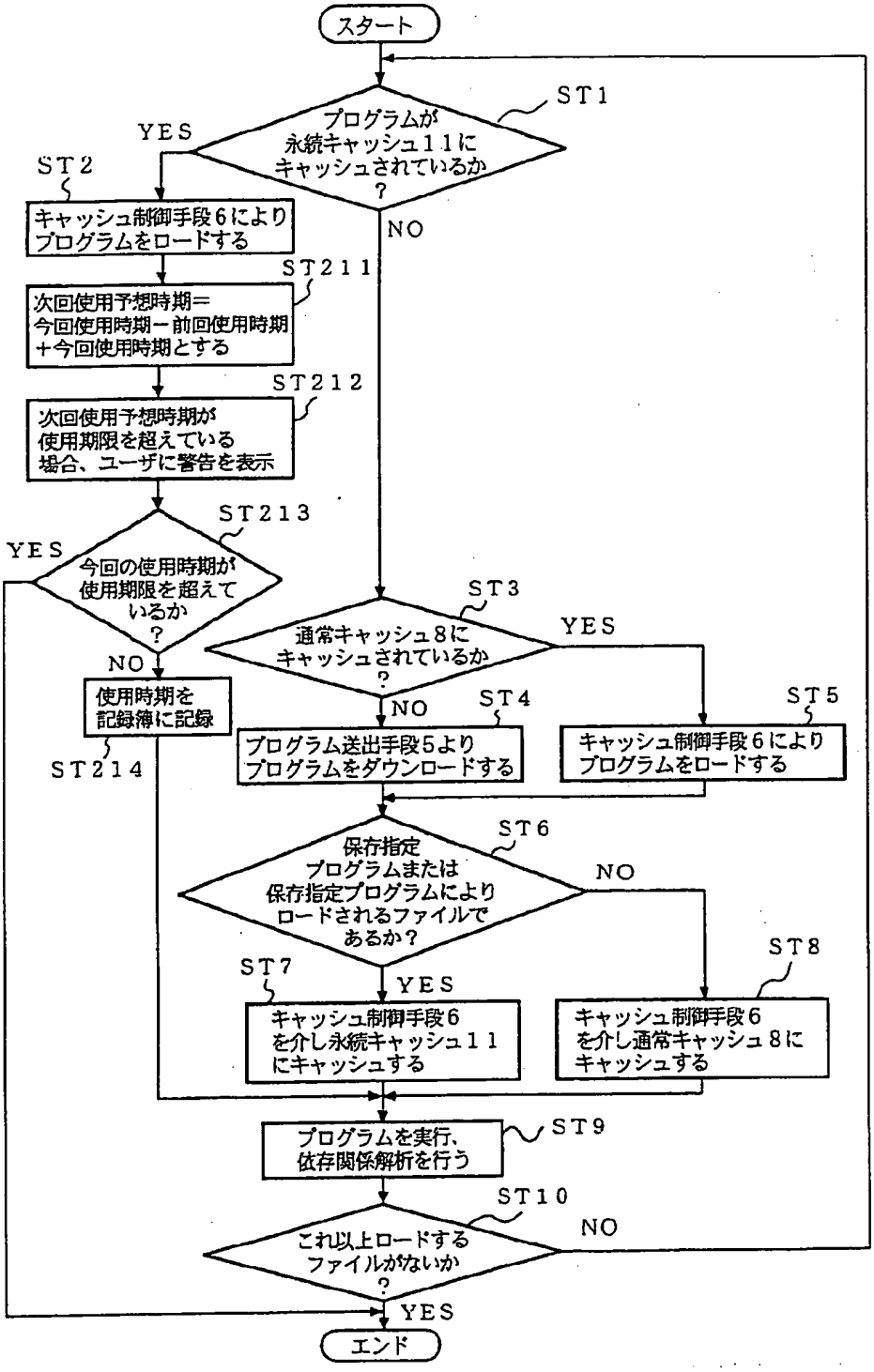
【図１６】


スタート
ST1
プログラムが永続キャッシュ11にキャッシュされているか？
YES
NO
ST2
キャッシュ制御手段6bによりプログラムをロードする
ST3
通常キャッシュ8にキャッシュされているか？
YES
NO
ST4
プログラム送出手段5よりプログラムをダウンロードする
ST5
キャッシュ制御手段6bによりプログラムをロードする
ST6
保存指定プログラムまたは保存指定プログラムによりロードされるファイルであるか？
YES
NO
ST7
キャッシュ制御手段6bを介し永続キャッシュ11にキャッシュする
ST8
キャッシュ制御手段6bを介し通常キャッシュ8にキャッシュする
ST9
プログラムを実行、依存関係解析を行う
ST10
これ以上ロードするファイルがないか？
YES
NO
エンド
ST15
ネットワークに接続されているか？
YES
NO
ST16
更新チェック手段がプログラム送出手段よりプログラムをダウンロードする
ST17
更新チェック手段はダウンロードしたプログラムとキャッシュからロードした内容を比較する
ST18
更新チェック手段は変更があればユーザに知らせる

【図18】


スタート
ST1
プログラムが永続キャッシュ11にキャッシュされているか？
YES
NO
ST2
キャッシュ制御手段6cによりプログラムをロードする
ST3
通常キャッシュ8にキャッシュされているか？
YES
NO
ST4
プログラム送出手段5よりプログラムをダウンロードする
ST5
キャッシュ制御手段6bによりプログラムをロードする
ST6
保存指定プログラムまたは保存指定プログラムによりロードされるファイルであるか？
YES
NO
ST7
キャッシュ制御手段6cを介し永続キャッシュ11にキャッシュする
ST8
キャッシュ制御手段6cを介し通常キャッシュ8にキャッシュする
ST9
プログラムを実行、依存関係解析を行う
ST10
これ以上ロードするファイルがないか？
YES
NO
エンド
ST15
ネットワークに接続されているか？
YES
NO
ST16
更新チェック・ロード手段がプログラム送出手段よりプログラムをダウンロードする
ST17
更新チェック・ロード手段はダウンロードしたプログラムとキャッシュからロードした内容を比較する
ST18
更新チェック手段は変更があればユーザに知らせる
ST19
ユーザがプログラム更新を指示したか？
YES
NO
ST20
更新チェック・ロード手段は比較結果より変更されているプログラムをロードし、さらにそのファイルから依存解析を行ってプログラムを更新する

【図２１】


スタート
ST1
プログラムが永続キャッシュ１１にキャッシュされているか？
YES
NO
ST2
キャッシュ制御手段６によりプログラムをロードする
ST211
次回使用予想時期＝今回使用時期－前回使用時期＋今回使用時期とする
ST212
次回使用予想時期が使用期限を超えている場合、ユーザに警告を表示
ST213
今回の使用時期が使用期限を超えているか？
YES
NO
使用時期を記録簿に記録
ST214
ST3
通常キャッシュ８にキャッシュされているか？
YES
NO
ST4
プログラム送出手段５よりプログラムをダウンロードする
ST5
キャッシュ制御手段６によりプログラムをロードする
ST6
保存指定プログラムまたは保存指定プログラムによりロードされるファイルであるか？
NO
YES
ST7
キャッシュ制御手段６を介し永続キャッシュ１１にキャッシュする
ST8
キャッシュ制御手段６を介し通常キャッシュ８にキャッシュする
ST9
プログラムを実行、依存関係解析を行う
ST10
これ以上ロードするファイルがないか？
NO
YES
エンド